A New Foundation of Attack Trees in Linear Logic

Harley Eades III

Computer Science
Augusta University
harley.eades@gmail.com

Abstract. In this short paper I provide an update on an ongoing research project investigating founding attack trees in the resource conscious theory called linear logic. I introduce a new linear logic called the Attack Tree Linear Logic (ATLL) which models attack trees as formulas and uses implication to reason about attack trees. I then introduce a new generalization of Bucciarelli and Ehrhards indexed linear logic to obtain Indexed ATLL which adds costs to attack trees. The most interesting aspect of Indexed ATLL is that implication is endowed with a relation for taking into consideration the costs while reasoning about attack trees.

1 Introduction

What is a mathematical model of attack trees? There have been numerous proposed answers to this question. Some examples are propositional logic, multisets, directed acyclic graphs, source sink graphs (or parallel-series pomsets), Petri nets, and Markov processes. Is there a unifying foundation in common to each of these proposed models? Furthermore, can this unifying foundation be used to further the field of attack trees and build new tools for conducting threat analysis?

The answer to the first question is positive, but the answer to the second question is open. Each of the proposed models listed above have something in common. They can all be modeled in some form of a symmetric monoidal category [9,1,3,4]. That is all well and good, but what can we gain from monoidal categories?

Monoidal categories are a mathematical model of linear logic as observed through the beautiful Curry-Howard-Lambek correspondence [7]. In linear logic every hypothesis must be used exactly once, and hence, if we view a hypothesis as a resource, then this property can be stated as every resource must be consumed. This linearity property is achieved by removing the structural rules for weakening and contraction from classical or intuitionistic logic.

Multisets and Petri nets both capture the idea that the nodes of an attack tree consist of the attack action and the state – the resource – of the system being analyzed. As it turns out, linear logic has been shown to be a logical foundation for multisets [9] and Petri Nets [1]. Thus, linear logic has the ability to model the state as well as attack actions of the goals of an attack tree.

In this short paper I give an update of an ongoing project[1] whose main goal is to determine the suitability of linear logic as a foundation for attack trees; the type of attack trees considered in this paper are attack trees with sequential conjunction due to Jhawar et al. [6]. I introduce a new logical foundation of attack trees with and without costs in linear logic.

This new linear logic is called the Attack Tree Linear Logic (ATLL) (Section 3). Attack trees are modeled in ATLL as linear formulas where base attacks are atomic formulas, and each branching node corresponds to a binary logical connective. Then reasoning about attack trees corresponds to proving implications between attack trees. In fact, every equation of attack trees from Jhawar et al.'s work on attack trees with sequential conjunction [6] can be proven as an implication in ATLL. In addition, implication can be used to prove subattack tree relationships between attack trees.

Using a generalization of Bucciarelli and Ehrhards [2] indexed linear logic I extend ATLL with costs. I call this extension Indexed ATLL (Section 4). In this system every base attack is annotated with a cost, and every branching node is annotated with a binary operation used for computing the cost at that node. Then implication is annotated with a binary relation on costs. This new binary relation is used to insure that the costs of two trees are related when proving implications between attack trees. Thus, one can prove facts about attack trees in ATLL that require one to reason about both the tree structure as well quantitative data in the tree simultaneously.

2 A Brief Introduction to Ordered Linear Logic

Before introducing ATLL and Indexed ATLL and how it can be used to model and reason about attack trees I first give a brief introduction to (ordered) linear logic and how it can be used to model various structures in computer science. I assume very little of the reader, and start with the basics of the specification of logics. Ordered linear logic (OLL) [8] will be the ongoing example throughout this section, because it is the logic ATLL is based on.

Every logic presented in this paper is in the form of sequent-style natural deduction. This type of formalization begins with the specifying the syntax of formulas and sequents. The latter is defined by a set of inference rules. The syntax for OLL is as follows:

$$\begin{array}{rr} \text{(formulas)} & A, B, C ::= b \mid A \odot B \mid A \rhd B \mid A \multimap B \\ \text{(ordered contexts)} & \Delta ::= \cdot \mid A \mid \Gamma, \Delta \\ \text{(unordered contexts)} & \Gamma ::= \cdot \mid A \mid \Theta, \Psi \end{array}$$

Formulas of OLL consist of atomic formulas denoted by b, a symmetric tensor product denoted by $A \odot B$, a non-symmetric tensor product denoted by $A \rhd B$ – in

[1] This material is based upon work supported by the National Science Foundation CRII CISE Research Initiation grant, "CRII:SHF: A New Foundation for Attack Trees Based on Monoidal Categories", under Grant No. 1565557.

ATLL the former will be used to model sequential conjunction, and the latter to model parallel conjunction – and linear implication denoted by $A \multimap B$. Contexts are lists of hypothesis where Δ is a list of ordered hypothesis, and Γ is a list of unordered hypothesis. The former are associated with the non-symmetric tensor product, and the latter with the symmetric tensor product. We denote the empty context by $\cdot$, and appending of contexts Γ_1 and Γ_2 by Γ_1, Γ_2.

Sequents of OLL are denoted by $\Delta; \Gamma \vdash A$. We say that the sequent $\Delta; \Gamma \vdash A$ holds if and only if given the hypothesis in Δ and Γ one can construct a proof of the formula A using the following inference rules:

$$\frac{}{\cdot; A \vdash A}\,\mathrm{id}_{\odot} \qquad \frac{}{A; \cdot \vdash A}\,\mathrm{id}_{\rhd} \qquad \frac{\Delta_1; \Gamma_1 \vdash A \quad \Delta_2; \Gamma_2 \vdash B}{\Delta_1, \Delta_2; \Gamma_1, \Gamma_2 \vdash A \odot B}\,\odot_I$$

$$\frac{\Delta_1; \Gamma_2 \vdash A \odot B \quad \Delta_2; \Gamma_1, A, B, \Gamma_3 \vdash C}{\Delta_1, \Delta_2; \Gamma_1, \Gamma_2, \Gamma_3 \vdash C}\,\odot_E \qquad \frac{\Delta_1; \Gamma_1 \vdash A \quad \Delta_2; \Gamma_2 \vdash B}{\Delta_1, \Delta_2; \Gamma_1, \Gamma_2 \vdash A \rhd B}\,\rhd_I$$

$$\frac{\Delta_2; \Gamma_1 \vdash A \rhd B \quad \Delta_1, A, B, \Delta_3; \Gamma_2 \vdash C}{\Delta_1, \Delta_2, \Delta_3; \Gamma_1, \Gamma_2 \vdash C}\,\rhd_E \qquad \frac{\Delta; \Gamma_1, A, B, \Gamma_2 \vdash C}{\Delta; \Gamma_1, B, A, \Gamma_2 \vdash C}\,\mathrm{sym}_{\odot}$$

$$\frac{\Delta; \Gamma, A \vdash B}{\Delta; \Gamma \vdash A \multimap B}\,\multimap_I \qquad \frac{\Delta_1; \Gamma_1 \vdash A \multimap B \quad \Delta_2; \Gamma_2 \vdash A}{\Delta_1, \Delta_2; \Gamma_1, \Gamma_2 \vdash B}\,\multimap_E$$

An inference rule should be read from top to bottom as an implication. The sequents on top of the line are the premises, and the sequent below the line is the conclusion. For example, one should read the rule $\odot_I$ as if $\Delta_1; \Gamma_1 \vdash A$ and $\Delta_2; \Gamma_2 \vdash B$ both hold, then $\Delta_1, \Delta_2; \Gamma_1, \Gamma_2 \vdash A \odot B$ holds. An inference rule with no premises are called axioms. For example, the rules $\mathrm{id}_{\odot}$ and $\mathrm{id}_{\rhd}$ are both axioms.

Inference rules are read from top to bottom, but applied from bottom to top. A sequent, $\Delta; \Gamma \vdash A$, holds if and only if there is a series of inference rules that can be composed into a derivation. Every derivation is a tree where the root is the sequent we wish to prove, and then rules are composed to form a tree. A rule can be composed with another if the formers conclusion matches a premise of the latter. If every branch of this tree reaches an axiom, then it is a valid proof, but if any branch gets stuck, then the sequent does not hold. The following is a proof that the symmetric tensor is indeed symmetric:

$$\dfrac{\dfrac{\dfrac{}{\cdot; A \odot B \vdash A \odot B}\,\mathrm{ID}_{\odot} \qquad \dfrac{\dfrac{\dfrac{}{\cdot; B \vdash B}\,\mathrm{ID}_{\odot} \qquad \dfrac{}{\cdot; A \vdash A}\,\mathrm{ID}_{\odot}}{\cdot; B, A \vdash B \odot A}\,\odot_I}{\cdot; A, B \vdash B \odot A}\,\mathrm{SYM}_{\odot}}{\cdot; A \odot B \vdash B \odot A}\,\odot_E}{\cdot; \cdot \vdash (A \odot B) \multimap (B \odot A)}\,\multimap_I$$

We call the root of the tree the goal of the proof, and as we construct a derivation this goal is refined into potentially several new subgoals. This is called goal

directed proof. As we can see the previous derivation is a valid proof of the goal $\cdot;\cdot \vdash (A \odot B) \multimap (B \odot A)$.

The following derivation is an invalid proof that the non-symmetric tensor product is symmetric:

$$\dfrac{\dfrac{\dfrac{}{\cdot; A \rhd B \vdash A \rhd B}\,\mathrm{ID}_{\odot} \qquad A, B;\cdot \vdash B \rhd A}{\cdot; A \rhd B \vdash B \rhd A}\,\rhd_E}{\cdot;\cdot \vdash (A \rhd B) \multimap (B \rhd A)}\,\multimap_I$$

At this point we have refined our goal to the subgoal $A, B;\cdot \vdash B \rhd A$, but this is impossible to prove, because the exchange rule $\mathrm{sym}_{\odot}$ cannot be be applied to the ordered context to commute A and B.

Building proofs from the inference rules given above can often be tedious and long. To make this process easier it is often necessary to introduce new rules that can be proven to be valid in terms of the inference rules already given. Rules introduced in this way are called derivable inference rules. One derivable rule we will make use of is the following:

$$\dfrac{\Delta_2; \Gamma_2 \vdash A \multimap B \quad \Delta_1; \Gamma_1 \vdash B \multimap C}{\Delta_1, \Delta_2; \Gamma_1, \Gamma_2 \vdash A \multimap C}\,comp$$

Proving this rule is derivable amounts to creating a derivation whose branches terminate at either an axiom or the premises $\Delta_2; \Gamma_2 \vdash A \multimap B$ or $\Delta_1; \Gamma_1 \vdash B \multimap C$. The proof is as follows:

$$\dfrac{\dfrac{\Delta_1; \Gamma_1 \vdash B \multimap C \qquad \dfrac{\Delta_2; \Gamma_2 \vdash A \multimap B \qquad \dfrac{}{\cdot; A \vdash A}\,\mathrm{ID}_{\odot}}{\Delta_2; \Gamma_2, A \vdash B}\,\multimap_E}{\Delta_1, \Delta_2; \Gamma_1, \Gamma_2, A \vdash C}\,\multimap_E}{\Delta_1, \Delta_2; \Gamma_1, \Gamma_2 \vdash A \multimap C}\,\multimap_I$$

At this point I have introduced the basics of sequent-style natural deduction, but nothing I have said up to now has been specific to ordered linear logic. We call the this logic ordered, because it contains the non-symmetric tensor product. Its non-symmetry is enforced by the separation of hypotheses. What makes a logic linear?

Linear logic was first introduced by Girard [5] in the late eighties. A logic is linear if every hypothesis is used exactly once. Thus, no hypothesis can be duplicated or removed at will, that is, the structural rules for contraction (duplication) and weakening (removal) must be banned from the logic. This property makes linear logic particularly suited for reasoning about state-based systems.

We enforce this property by restricting the inference rules in two ways. First, consider the identity axioms for OLL:

$$\dfrac{}{\cdot; A \vdash A}\,\mathrm{id}_{\odot} \qquad\qquad \dfrac{}{A;\cdot \vdash A}\,\mathrm{id}_{\rhd}$$

The contexts are restricted to containing at most the hypothesis the axiom is proving holds. The other restriction is when applying the inference rules during the construction of derivations one is not allowed to copy or introduce new hypothesis across premises. This is why we take great care at separating the contexts in the definition of the inference rules.

Modeling structures in logic corresponds to interpreting the structures as formulas, and then reasoning about the structures corresponds to proving implications between the interpretations. As we will see, attack trees will be modeled as formulas of linear logic, and then we will prove several properties of attack trees by proving implications between them.

3 Attack Trees without Costs

In this section I introduce attack trees with sequential conjunction – sometimes referred to SAND attack trees – without cost annotations in the hope that doing so will make the main ideas of this work more clear to the reader. This formulation of attack trees was first proposed by Jhawar et al. [6]. First, I define the syntax of attack trees then a linear logic for reasoning about them.

Definition 1. *Suppose* B *is a set of base attacks whose elements are denoted by* b*. Then an* ***attack tree*** *is defined by the following grammar:*

$$T ::= b \mid T_1 \odot T_2 \mid T_1 \sqcup T_2 \mid T_1 \rhd T_2$$

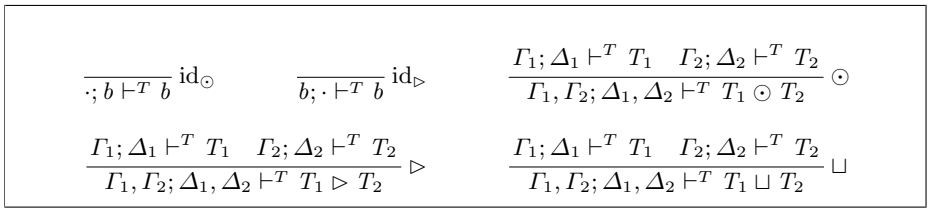
I denote unsynchronized parallel conjunction of attacks by $T_1 \odot T_2$*, choice between attacks by* $T_1 \sqcup T_2$*, and sequential conjunction of attacks by* $T_1 \rhd T_2$*.*

Now we define the Attack Tree Linear Logic (ATLL) for reasoning about attack trees. In ATLL attack trees are modeled as formulas, and thus, reasoning about attack trees should correspond to proving implications between attack trees. Furthermore, some care must be taken so that choice and parallel conjunction are symmetric, but sequential conjunction is not. We enforce this property in ATLL by separating hypotheses as is usually done in ordered linear logic [8].

Definition 2. *Suppose* B *is a set of base attacks whose elements are denoted by* b*. The syntax of ATLL formulas and contexts are defined as follows:*

$$\begin{aligned} \textit{(formulas)}\ E &::= b \mid E_1 \odot E_1 \mid E_1 \sqcup E_2 \mid E_1 \rhd E_2 \mid E_1 \multimap E_2 \\ \textit{(base contexts)}\ \Gamma, \Delta &::= \cdot \mid b \mid \Gamma, \Delta \\ \textit{(general contexts)}\ \Theta, \Psi &::= \cdot \mid E \mid \Gamma, \Delta \end{aligned}$$

ATLL formulas are defined as an extension of the syntax for attack trees with linear implication. An attack tree then corresponds to an ATLL formula without implication. Suppose E is an ATLL formula such that every base attack in E which is a leaf of sequential conjunction is in the base context Γ, and every leaf of parallel conjunction is in the base context Δ. Then E is an attack tree if and only if $\Gamma; \Delta \vdash^{T} E$ holds with respect to the rules in Figure 1.

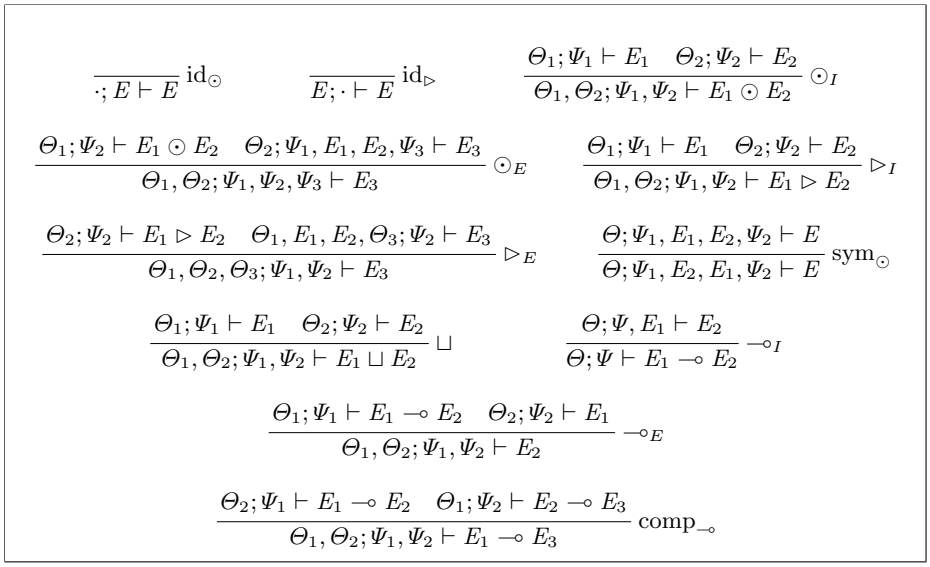


$$\frac{}{\cdot; b \vdash^T b}\,\mathrm{id}_{\odot} \qquad \frac{}{b; \cdot \vdash^T b}\,\mathrm{id}_{\triangleright} \qquad \frac{\Gamma_1; \Delta_1 \vdash^T T_1 \quad \Gamma_2; \Delta_2 \vdash^T T_2}{\Gamma_1, \Gamma_2; \Delta_1, \Delta_2 \vdash^T T_1 \odot T_2}\,\odot$$

$$\frac{\Gamma_1; \Delta_1 \vdash^T T_1 \quad \Gamma_2; \Delta_2 \vdash^T T_2}{\Gamma_1, \Gamma_2; \Delta_1, \Delta_2 \vdash^T T_1 \triangleright T_2}\,\triangleright \qquad \frac{\Gamma_1; \Delta_1 \vdash^T T_1 \quad \Gamma_2; \Delta_2 \vdash^T T_2}{\Gamma_1, \Gamma_2; \Delta_1, \Delta_2 \vdash^T T_1 \sqcup T_2}\,\sqcup$$

Fig. 1. Well Formed Attack Trees in ATLL

The inference rules in Figure 1 enforce several properties. First, they enforce that every base attack is used exactly once, and that an attack tree does not mention linear implication. Furthermore, they allow us to restrict our reasoning to attack trees which I conjecture will result in our reasoning being more amenable to automation.



$$\frac{}{\cdot; E \vdash E}\,\mathrm{id}_{\odot} \qquad \frac{}{E; \cdot \vdash E}\,\mathrm{id}_{\triangleright} \qquad \frac{\Theta_1; \Psi_1 \vdash E_1 \quad \Theta_2; \Psi_2 \vdash E_2}{\Theta_1, \Theta_2; \Psi_1, \Psi_2 \vdash E_1 \odot E_2}\,\odot_I$$

$$\frac{\Theta_1; \Psi_2 \vdash E_1 \odot E_2 \quad \Theta_2; \Psi_1, E_1, E_2, \Psi_3 \vdash E_3}{\Theta_1, \Theta_2; \Psi_1, \Psi_2, \Psi_3 \vdash E_3}\,\odot_E \qquad \frac{\Theta_1; \Psi_1 \vdash E_1 \quad \Theta_2; \Psi_2 \vdash E_2}{\Theta_1, \Theta_2; \Psi_1, \Psi_2 \vdash E_1 \triangleright E_2}\,\triangleright_I$$

$$\frac{\Theta_2; \Psi_2 \vdash E_1 \triangleright E_2 \quad \Theta_1, E_1, E_2, \Theta_3; \Psi_2 \vdash E_3}{\Theta_1, \Theta_2, \Theta_3; \Psi_1, \Psi_2 \vdash E_3}\,\triangleright_E \qquad \frac{\Theta; \Psi_1, E_1, E_2, \Psi_2 \vdash E}{\Theta; \Psi_1, E_2, E_1, \Psi_2 \vdash E}\,\mathrm{sym}_{\odot}$$

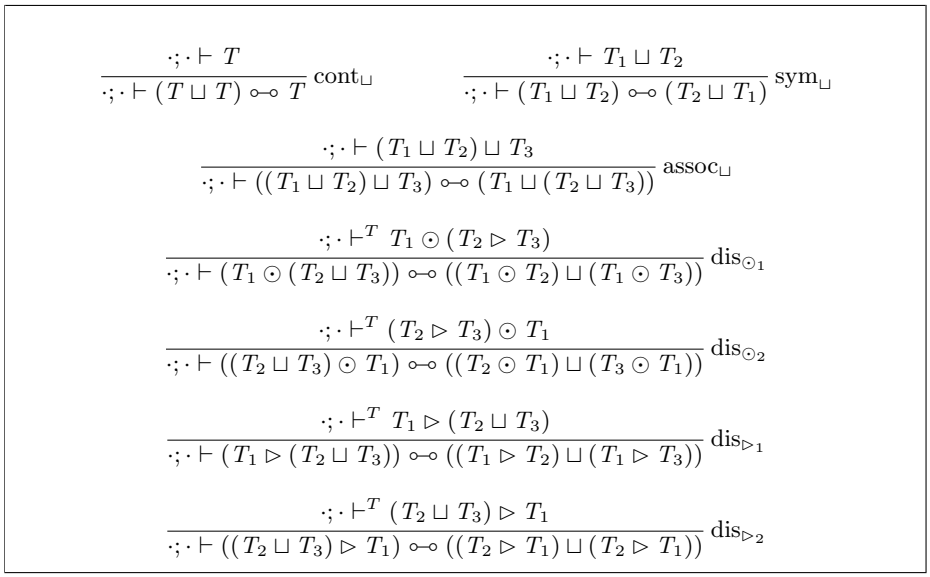
$$\frac{\Theta_1; \Psi_1 \vdash E_1 \quad \Theta_2; \Psi_2 \vdash E_2}{\Theta_1, \Theta_2; \Psi_1, \Psi_2 \vdash E_1 \sqcup E_2}\,\sqcup \qquad \frac{\Theta; \Psi, E_1 \vdash E_2}{\Theta; \Psi \vdash E_1 \multimap E_2}\,\multimap_I$$

$$\frac{\Theta_1; \Psi_1 \vdash E_1 \multimap E_2 \quad \Theta_2; \Psi_2 \vdash E_1}{\Theta_1, \Theta_2; \Psi_1, \Psi_2 \vdash E_2}\,\multimap_E$$

$$\frac{\Theta_2; \Psi_1 \vdash E_1 \multimap E_2 \quad \Theta_1; \Psi_2 \vdash E_2 \multimap E_3}{\Theta_1, \Theta_2; \Psi_1, \Psi_2 \vdash E_1 \multimap E_3}\,\mathrm{comp}_{\multimap}$$

Fig. 2. ATLL Logical Inference Rules

The ATLL logical inference rules are defined in Figure 2. These rules make up a basic linear logic for attack trees. For example, we can prove that sequential and parallel conjunction are associative, and the latter is symmetric, but there are a number of facts that these rules do not prove. They cannot prove that choice is associative and symmetric, and they cannot prove the usual distributive laws associated with attack trees with sequential conjunction [6]. To overcome this limitation we add the attack tree axioms in Figure 3 to ATLL. The logical



$$\frac{\cdot;\cdot \vdash T}{\cdot;\cdot \vdash (T \sqcup T) \multimap T}\ \mathrm{cont}_{\sqcup} \qquad \frac{\cdot;\cdot \vdash T_1 \sqcup T_2}{\cdot;\cdot \vdash (T_1 \sqcup T_2) \multimap (T_2 \sqcup T_1)}\ \mathrm{sym}_{\sqcup}$$

$$\frac{\cdot;\cdot \vdash (T_1 \sqcup T_2) \sqcup T_3}{\cdot;\cdot \vdash ((T_1 \sqcup T_2) \sqcup T_3) \multimap (T_1 \sqcup (T_2 \sqcup T_3))}\ \mathrm{assoc}_{\sqcup}$$

$$\frac{\cdot;\cdot \vdash^{T} T_1 \odot (T_2 \rhd T_3)}{\cdot;\cdot \vdash (T_1 \odot (T_2 \sqcup T_3)) \multimap ((T_1 \odot T_2) \sqcup (T_1 \odot T_3))}\ \mathrm{dis}_{\odot_1}$$

$$\frac{\cdot;\cdot \vdash^{T} (T_2 \rhd T_3) \odot T_1}{\cdot;\cdot \vdash ((T_2 \sqcup T_3) \odot T_1) \multimap ((T_2 \odot T_1) \sqcup (T_3 \odot T_1))}\ \mathrm{dis}_{\odot_2}$$

$$\frac{\cdot;\cdot \vdash^{T} T_1 \rhd (T_2 \sqcup T_3)}{\cdot;\cdot \vdash (T_1 \rhd (T_2 \sqcup T_3)) \multimap ((T_1 \rhd T_2) \sqcup (T_1 \rhd T_3))}\ \mathrm{dis}_{\rhd_1}$$

$$\frac{\cdot;\cdot \vdash^{T} (T_2 \sqcup T_3) \rhd T_1}{\cdot;\cdot \vdash ((T_2 \sqcup T_3) \rhd T_1) \multimap ((T_2 \rhd T_1) \sqcup (T_2 \rhd T_1))}\ \mathrm{dis}_{\rhd_2}$$

Fig. 3. ATLL Attack Tree Axioms

connective $\multimap$ is linear biimplication which is defined in the usual way as classical implication. The reader should note that these axioms are restricted to attack trees only and not applicable to full ATLL formulas. Furthermore, the reader should note that the rule $\mathrm{comp}_{\multimap}$ is very useful when proving properties of attack trees, but is actually an admissible rule, meaning, it can be proven to hold in terms of the other rules in ATLL, but to simplify the presentation given here it is taken as part of the system. In addition, the rule $\mathrm{dis}_{\odot_2}$ is admissible as well, because parallel conjunction is symmetric.

The ATLL logical inference rules also make it clear why we separate hypotheses into two contexts. The exchange rule, $\mathrm{sym}_{\odot}$, allows one to exchange hypotheses in the context Ψ, but notice that there is no exchange rule for the context Θ. Thus, one should think of the hypotheses in Ψ as running in parallel and the hypotheses in Θ as running sequentially. In fact, this interpretation is reinforced by the elimination rules, $\odot_E$ and $\rhd_E$, for parallel and sequential conjunction respectively.

Finally, ATLL allows one to prove every equivalence for attack trees with sequential conjunction given by Jhawar et al. [6] as linear implications. I conjecture that it should be possible to define a proof search algorithm for ATLL to automate equational reasoning on attack trees, but this is future work. However, using implication as a means to reason about attack trees opens the door for different types of reasoning.

Suppose T_1 is a larger attack tree, and we wish to know if the attack tree T_2 is contained within T_1, that is, T_2 is a subattack tree of T_1. Currently, ATLL will not allow us to prove this, but it can if we add the following rules:

$$\frac{\Theta_1, \Theta_2; \Psi \vdash E \quad \Gamma; \Delta \vdash^T T}{\Theta_1, T, \Theta_2; \Psi \vdash E}\ \mathrm{wk}_{\odot} \qquad \frac{\Theta; \Psi_1, \Psi_2 \vdash E \quad \Gamma; \Delta \vdash^T T}{\Theta; \Psi_1, T, \Psi_2 \vdash E}\ \mathrm{wk}_{\rhd}$$

These rules allow us to discard parts of a larger attack tree while constructing proof derivations, but this only works for attack trees not general ATLL formulas. Thus, with these rules ATLL becomes slightly affine. We can now use these rules to determine if $T_1 \multimap T_2$ holds, and if it does, then T_2 is a subattack tree of T_1.

I now give a couple of examples to illustrate using the system to prove properties of attack trees. Consider the two example trees used to introduce the SPTool for checking equivalence of attack trees by Krody et al. [**?**]:

$$\begin{aligned} T_1 &:= (B_1 \odot (B_2 \sqcup B_3)) \rhd B_4 \\ T_2 &:= ((B_1 \odot B_2) \rhd B_4) \sqcup ((B_1 \odot B_3) \rhd B_4) \end{aligned}$$

where B_i for $1 \leq i \leq 4$ are the base attacks. The tree T_1 corresponds to the ATM attack tree from Figure 1 of [**?**], and T_2 corresponds to the canonical tree given in Figure 2 of [**?**]. The attack trees T_1 and T_2 are equivalent if and only if $T_1 \multimap\!\!\circ\!\!- T_2$ holds. First, ATLL attack tree axioms are biimplications, and hence, we may treat these axioms as equivalences. For example, suppose $\cdot; \cdot \vdash E_1 \multimap\!\!\circ\!\!- E_2$ and $\Theta; \Psi \vdash E_1 \rhd E_3$ both hold, then we can prove that $\Theta; \Psi \vdash E_2 \rhd E_3$ holds:

$$\dfrac{\Theta; \Psi \vdash E_1 \rhd E_3 \qquad \dfrac{\dfrac{\cdot; \cdot \vdash E_1 \multimap E_2 \qquad \dfrac{}{E_1; \cdot \vdash E_1}}{E_1; \cdot \vdash E_2}\ \multimap_E \qquad \dfrac{}{E_3; \cdot \vdash E_3}}{E_1, E_3; \cdot \vdash E_2 \rhd E_3}\ \rhd_E}{\Theta; \Psi \vdash E_2 \rhd E_3}\ \rhd_E$$

Thus, using this realization and the ATLL attack tree axioms we can prove that T_1 is equivalent to T_2 informally as follows:

$$\begin{aligned} T_1 :=\ & (B_1 \odot (B_2 \sqcup B_3)) \rhd B_4 && \text{(definition)} \\ \multimap\!\!\circ\!\!-\ & ((B_1 \odot B_2) \sqcup (B_1 \odot B_3)) \rhd B_4 && (\text{rule } \mathrm{dis}_{\odot_1}) \\ \multimap\!\!\circ\!\!-\ & ((B_1 \odot B_2) \rhd B_4) \sqcup ((B_1 \odot B_3) \rhd B_4) && (\text{rule } \mathrm{dis}_{\rhd_2}) \\ :=\ & T_2 && \text{(definition)} \end{aligned}$$

The corresponding formal derivation is quite large, but is easily constructible using the previous reasoning.

4 Attack Trees with Costs

Now I generalize Bucciarelli and Ehrhards indexed linear logic [2] to obtain an extension of ATLL called Indexed ATLL with the ability to model and reason about attack trees with costs.

Definition 3. *Suppose* B *is a set of base attacks whose elements are denoted by* b*, and* C *is a set of costs whose elements are denoted by* c*. Additionally, let* $(\mathsf{C}, \mathsf{op}_\odot)$ *and* $(\mathsf{C}, \mathsf{op}_\sqcup)$ *be symmetric monoids on* C*, and* $(\mathsf{C}, \mathsf{op}_\rhd)$ *be a monoid on* C*, where* $\mathsf{op}_\odot$ *and* $\mathsf{op}_\rhd$ *distribute over* $\mathsf{op}_\sqcup$*. Then an* ***attack tree with costs*** *is defined by the following grammar:*

$$T ::= (b, c) \mid T_1 \odot T_2 \mid T_1 \sqcup T_2 \mid T_1 \rhd T_2$$

$$\frac{}{\cdot; (b, c) \vdash^T_c b}\ \mathrm{id}_\odot \qquad \frac{}{(b, c); \cdot \vdash^T_c b}\ \mathrm{id}_\rhd \qquad \frac{\Gamma_1; \Delta_1 \vdash^T_{c_1} T_1 \quad \Gamma_2; \Delta_2 \vdash^T_{c_2} T_2}{\Gamma_1, \Gamma_2; \Delta_1, \Delta_2 \vdash^T_{\mathsf{op}_\odot(c_1, c_2)} T_1 \odot T_2}\ \odot$$

$$\frac{\Gamma_1; \Delta_1 \vdash^T_{c_1} T_1 \quad \Gamma_2; \Delta_2 \vdash^T_{c_2} T_2}{\Gamma_1, \Gamma_2; \Delta_1, \Delta_2 \vdash^T_{\mathsf{op}_\rhd(c_1, c_2)} T_1 \rhd T_2}\ \rhd \qquad \frac{\Gamma_1; \Delta_1 \vdash^T_{c_1} T_1 \quad \Gamma_2; \Delta_2 \vdash^T_{c_2} T_2}{\Gamma_1, \Gamma_2; \Delta_1, \Delta_2 \vdash^T_{\mathsf{op}_\sqcup(c_1, c_2)} T_1 \sqcup T_2}\ \sqcup$$

Fig. 4. Well Formed Attack Trees in Indexed ATLL

Each base attack is annotated with a cost, and every branching node is associated with a binary operation for computing the cost at that node using the costs of the left subtree and the right subtree.

I now extend the syntax and rules of ATLL into Indexed ATLL, but due to space I do not give every rule here, but only the most interesting ones, because they are similar to ATLL; the entire system is defined in Appendix A.

Definition 4. *Suppose* B *is a set of base attacks whose elements are denoted by* b*, and* C *is a set of costs whose elements are denoted by* c*. Additionally, let* $(\mathsf{C}, \mathsf{op}_\odot)$ *and* $(\mathsf{C}, \mathsf{op}_\sqcup)$ *be symmetric monoids on* C*, and* $(\mathsf{C}, \mathsf{op}_\rhd)$ *be a monoid on* C*, where* $\mathsf{op}_\odot$ *and* $\mathsf{op}_\rhd$ *distribute over* $\mathsf{op}_\sqcup$*. Furthermore, let* $(\mathsf{C}, \mathsf{rel}_{\multimap})$ *be a preorder on* C*. Then the syntax of Indexed ATLL formulas and contexts are defined as follows:*

$$\begin{aligned} \textit{(formulas)}\ E &::= (b, c) \mid E_1 \odot E_1 \mid E_1 \sqcup E_2 \mid E_1 \rhd E_2 \mid (E_1, c) \multimap E_2 \\ \textit{(base contexts)}\ \Gamma, \Delta &::= \cdot \mid (b, c) \mid \Gamma, \Delta \\ \textit{(general contexts)}\ \Theta, \Psi &::= \cdot \mid (E, c) \mid \Gamma, \Delta \end{aligned}$$

As we can see the only major changes to the ATLL syntax is that all hypotheses are annotated with costs.

The inference rules for well-formed attack trees in Indexed ATLL are given in Figure 4. The judgment is now denoted by $\Gamma; \Delta \vdash^T_c T$ where c is the fully computed cost of the root of the attack tree T. Notice that this cost is computed during the construction of a proof derivation. This cost will be used along with the relation on implication to influence the relationship between attack trees.

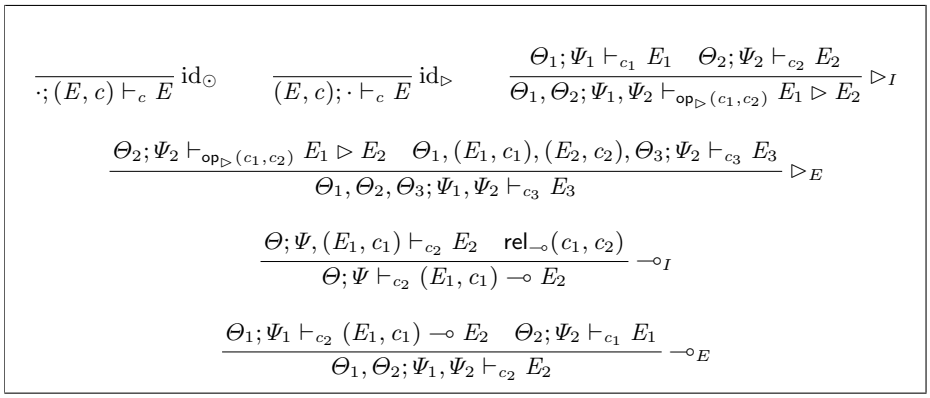


Fig. 5. Select Logical Inference Rules for Indexed ATLL

The logical inference rules for Indexed ATLL are given in Figure 5. As I mentioned above I only give a select few of the rules here, but the complete system can be found in Appendix A. The most interesting rule is the introduction rule for implication, $\multimap_I$, because the second premise is $\mathsf{rel}_{\multimap}(c_1, c_2)$.

Suppose (T_1, c_1) and (T_2, c_2) are two attack trees with the respective costs, and the relation $\mathsf{rel}_{\multimap}(c_1, c_2)$ holds if and only if $c_1 \leq c_2$. If we wanted to show that the attack tree T_1 is equivalent to the attack tree T_2, then we would have to show $T_1 \circ\!\!-\!\!\circ\ T_2$ where $\mathsf{rel}_{\circ\!-\!\circ}(c_1, c_2)$ holds if and only if $\mathsf{rel}_{\multimap}(c_1, c_2)$ and $\mathsf{rel}_{\multimap}(c_2, c_1)$ hold. This would require one to prove $\mathsf{rel}_{\circ\!-\!\circ}(c_1, c_2)$ which would require $c_1 = c_2$. Thus, in this example, attack trees would be equivalent if and only if they have the same logical structure and the same overall cost. Thus, several different types of reasoning on attack trees can be modeled in Index ATLL depending on the relation chosen for implication.

5 Conclusion and Future Work

I introduced a new linear logic for reasoning about attack trees called the Attack Tree Linear Logic (ATLL) with and without costs. Attack trees are modeled as logical formulas and proving relationships between attack trees corresponds to proving implications. Adding costs to ATLL results in a new indexed linear logic called Indexed ATLL. Then I showed how endowing implication with a relation on costs can be used to include the costs when reasoning about attack trees.

ATLL is just one part of this project, the following are some next steps with respect to (Indexed) ATLL and other parts of the project:

- Define a term assignment for (Indexed) ATLL that can be used to define attack trees using a functional programming language.
- Determine the applicability of automating reasoning on attack trees in (Indexed) ATLL using proof search.

– Finish studying the categorical model to which (Indexed) ATLL is based.
– Finishing the development of a modular and stand alone IDE for attack tree in Javascript that can be used with various formalizations of attack trees. Hopefully, the community will be able to make use of this interface.
– Develop the concrete implementation, called Lina, of (Indexed) ATLL with automation.

References

1. Carolyn Brown, Doug Gurr, and Valeria Paiva. A linear specification language for petri nets. *DAIMI Report Series*, 20(363), 1991.
2. Antonio Bucciarelli and Thomas Ehrhard. On phase semantics and denotational semantics in multiplicativeâadditive linear logic. *Annals of Pure and Applied Logic*, 102(3):247 – 282, 2000.
3. Marcelo Fiore and Marco Devesas Campos. *Computation, Logic, Games, and Quantum Foundations. The Many Facets of Samson Abramsky: Essays Dedicated to Samson Abramsky on the Occasion of His 60th Birthday*, chapter The Algebra of Directed Acyclic Graphs, pages 37–51. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.
4. Luisa Francesco Albasini, Nicoletta Sabadini, and Robert F. C. Walters. The compositional construction of markov processes. *Applied Categorical Structures*, 19(1):425–437, 2010.
5. Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50(1):1 – 101, 1987.
6. Ravi Jhawar, Barbara Kordy, Sjouke Mauw, SaÅ!'a RadomiroviÄ, and Rolando Trujillo-Rasua. Attack trees with sequential conjunction. In Hannes Federrath and Dieter Gollmann, editors, *ICT Systems Security and Privacy Protection*, volume 455 of *IFIP Advances in Information and Communication Technology*, pages 339–353. Springer International Publishing, 2015.
7. Paul-André Melliès. Categorical semantics of linear logic. In Pierre-Louis Curien, Hugo Herbelin, Jean-Louis Krivine, and Paul-André Melliès, editors, *Interactive models of computation and program behaviour*. Panoramas et Synthèses 27, Société Mathématique de France, 2009.
8. Jeff Polakow. *Ordered linear logic and applications*. PhD thesis, Carnegie Mellon University, 2001.
9. A Tzouvaras. The linear logic of multisets. *Logic Journal of IGPL*, 6(6):901–916, 1998.

Appendix

A The Full Specification of Indexed ATLL

Definition 4. *Suppose* B *is a set of base attacks whose elements are denoted by* b*, and* C *is a set of costs whose elements are denoted by* c*. Additionally, let* $(\mathsf{C}, \mathsf{op}_{\odot})$ *and* $(\mathsf{C}, \mathsf{op}_{\sqcup})$ *be symmetric monoids on* C*, and* $(\mathsf{C}, \mathsf{op}_{\triangleright})$ *be a monoid on* C*, where* $\mathsf{op}_{\odot}$ *and* $\mathsf{op}_{\triangleright}$ *distribute over* $\mathsf{op}_{\sqcup}$*. Furthermore, let* $(\mathsf{C}, \mathsf{rel}_{\multimap})$ *be a*

preorder on C*. Then the syntax of Indexed ATLL formulas and contexts are defined as follows:*

$$\begin{aligned}
\textit{(formulas)}\ E &::= (b, c) \mid E_1 \odot E_1 \mid E_1 \sqcup E_2 \mid E_1 \rhd E_2 \mid (E_1, c) \multimap E_2 \\
\textit{(base contexts)}\ \Gamma, \Delta &::= \cdot \mid (b, c) \mid \Gamma, \Delta \\
\textit{(general contexts)}\ \Theta, \Psi &::= \cdot \mid (E, c) \mid \Gamma, \Delta
\end{aligned}$$

$$\frac{}{\cdot; (b, c) \vdash^T_c b}\ \mathrm{id}_{\odot} \qquad \frac{}{(b, c); \cdot \vdash^T_c b}\ \mathrm{id}_{\rhd} \qquad \frac{\Gamma_1; \Delta_1 \vdash^T_{c_1} T_1 \quad \Gamma_2; \Delta_2 \vdash^T_{c_2} T_2}{\Gamma_1, \Gamma_2; \Delta_1, \Delta_2 \vdash^T_{\mathsf{op}_{\odot}(c_1, c_2)} T_1 \odot T_2}\ \odot$$

$$\frac{\Gamma_1; \Delta_1 \vdash^T_{c_1} T_1 \quad \Gamma_2; \Delta_2 \vdash^T_{c_2} T_2}{\Gamma_1, \Gamma_2; \Delta_1, \Delta_2 \vdash^T_{\mathsf{op}_{\rhd}(c_1, c_2)} T_1 \rhd T_2}\ \rhd \qquad \frac{\Gamma_1; \Delta_1 \vdash^T_{c_1} T_1 \quad \Gamma_2; \Delta_2 \vdash^T_{c_2} T_2}{\Gamma_1, \Gamma_2; \Delta_1, \Delta_2 \vdash^T_{\mathsf{op}_{\sqcup}(c_1, c_2)} T_1 \sqcup T_2}\ \sqcup$$

Table 1. Well Formed Attack Trees in Indexed ATLL

$$\frac{}{\cdot;(E,c)\vdash_c E}\,\mathrm{id}_{\odot} \qquad \frac{}{(E,c);\cdot\vdash_c E}\,\mathrm{id}_{\triangleright} \qquad \frac{\Theta_1;\Psi_1\vdash_{c_1} E_1 \quad \Theta_2;\Psi_2\vdash_{c_2} E_2}{\Theta_1,\Theta_2;\Psi_1,\Psi_2\vdash_{\mathsf{op}_{\odot}(c_1,c_2)} E_1\odot E_2}\,\odot_I$$

$$\frac{\Theta_1;\Psi_2\vdash_{\mathsf{op}_{\odot}(c_1,c_2)} E_1\odot E_2 \quad \Theta_2;\Psi_1,(E_1,c_1),(E_2,c_2),\Psi_3\vdash_{c_3} E_3}{\Theta_1,\Theta_2;\Psi_1,\Psi_2,\Psi_3\vdash_{c_3} E_3}\,\odot_E$$

$$\frac{\Theta_1;\Psi_1\vdash_{c_1} E_1 \quad \Theta_2;\Psi_2\vdash_{c_2} E_2}{\Theta_1,\Theta_2;\Psi_1,\Psi_2\vdash_{\mathsf{op}_{\triangleright}(c_1,c_2)} E_1\triangleright E_2}\,\triangleright_I$$

$$\frac{\Theta_2;\Psi_2\vdash_{\mathsf{op}_{\triangleright}(c_1,c_2)} E_1\triangleright E_2 \quad \Theta_1,(E_1,c_1),(E_2,c_2),\Theta_3;\Psi_2\vdash_{c_3} E_3}{\Theta_1,\Theta_2,\Theta_3;\Psi_1,\Psi_2\vdash_{c_3} E_3}\,\triangleright_E$$

$$\frac{\Theta;\Psi_1,(E_1,c_1),(E_2,c_2),\Psi_2\vdash_c E}{\Theta;\Psi_1,(E_2,c_2),(E_1,c_1),\Psi_2\vdash_c E}\,\mathrm{sym}_{\odot} \qquad \frac{\Theta_1;\Psi_1\vdash_{c_1} E_1 \quad \Theta_2;\Psi_2\vdash_{c_2} E_2}{\Theta_1,\Theta_2;\Psi_1,\Psi_2\vdash_{\mathsf{op}_{\sqcup}(c_1,c_2)} E_1\sqcup E_2}\,\sqcup$$

$$\frac{\Theta;\Psi,(E_1,c_1)\vdash_{c_2} E_2 \quad \mathsf{rel}_{\multimap}(c_1,c_2)}{\Theta;\Psi\vdash_{c_2}(E_1,c_1)\multimap E_2}\,\multimap_I$$

$$\frac{\Theta_1;\Psi_1\vdash_{c_2}(E_1,c_1)\multimap E_2 \quad \Theta_2;\Psi_2\vdash_{c_1} E_1}{\Theta_1,\Theta_2;\Psi_1,\Psi_2\vdash_{c_2} E_2}\,\multimap_E$$

Table 2. Logical Inference Rules for Indexed ATLL

$$\frac{\cdot;\cdot\vdash_c T}{\cdot;\cdot\vdash_c (T\sqcup T)\multimap T}\,\mathrm{cont}_{\sqcup} \qquad \frac{\cdot;\cdot\vdash_c T_1\sqcup T_2}{\cdot;\cdot\vdash_c (T_1\sqcup T_2)\multimap (T_2\sqcup T_1)}\,\mathrm{sym}_{\sqcup}$$

$$\frac{\cdot;\cdot\vdash_c (T_1\sqcup T_2)\sqcup T_3}{\cdot;\cdot\vdash_c ((T_1\sqcup T_2)\sqcup T_3)\multimap (T_1\sqcup (T_2\sqcup T_3))}\,\mathrm{assoc}_{\sqcup}$$

$$\frac{\cdot;\cdot\vdash_c^T T_1\odot(T_2\triangleright T_3)}{\cdot;\cdot\vdash_c (T_1\odot(T_2\sqcup T_3))\multimap((T_1\odot T_2)\sqcup(T_1\odot T_3))}\,\mathrm{dis}_{\odot_1}$$

$$\frac{\cdot;\cdot\vdash_c^T (T_2\triangleright T_3)\odot T_1}{\cdot;\cdot\vdash_c ((T_2\sqcup T_3)\odot T_1)\multimap((T_2\odot T_1)\sqcup(T_3\odot T_1))}\,\mathrm{dis}_{\odot_2}$$

$$\frac{\cdot;\cdot\vdash_c^T T_1\triangleright(T_2\sqcup T_3)}{\cdot;\cdot\vdash_c (T_1\triangleright(T_2\sqcup T_3))\multimap((T_1\triangleright T_2)\sqcup(T_1\triangleright T_3))}\,\mathrm{dis}_{\triangleright_1}$$

$$\frac{\cdot;\cdot\vdash_c^T (T_2\sqcup T_3)\triangleright T_1}{\cdot;\cdot\vdash_c ((T_2\sqcup T_3)\triangleright T_1)\multimap((T_2\triangleright T_1)\sqcup(T_2\triangleright T_1))}\,\mathrm{dis}_{\triangleright_2}$$

Table 3. Indexed ATLL Attack Tree Axioms